UNIVERSITY OF MARYLAND AT COLLEGE PARK

DEPARTMENT OF COMPUTER SCIENCE

July 17, 1997

Scientific Officer Code: 333
Gary M. Koob
Office of Naval Research
Ballston Tower One
800 North Quincy Street
Arlington, Virginia 22217-5660

Re: N00014-94-1-0661

Dear Dr. Koob:

Enclosed please find the annual technical report for the above-referenced grant. The report has been distributed as listed below. If you have any questions or concerns, please contact me at (301) 405-2729.

Sincerely,

Andrea V. Busada

Andrea V. Busada
Research Coordinator
High Performance Systems Software Laboratory

Administrative Grants Officer
Office of Naval Research
Resident Representative
101 Marietta Street, Suite 2805
Atlanta, Georgia 30323-0008
1 copy

Director, Naval Research Laboratory
Attn.: Code 2627
Washington, DC 20375
1 copy

Defense Technical Information Center
Building 5, Cameron Station
Alexandria, Virginia 22304-6145
2 copies

DTIC QUALITY INSPECTED 3

GRANT NUMBER: N00014-94-1-0661

FORM A2-2
AUGMENTATION AWARDS FOR SCIENCE & ENGINEERING RESEARCH TRAINING (AASERT) REPORTING FORM

The Department of Defense (DOD) requires certain information to evaluate the effectiveness of the AASERT program. By accepting this Grant Modification, which bestows the AASERT funds, the Grantee agrees to provide the information requested below to the Government's technical point of contact by each annual anniversary of the AASERT award date.

1. Grantee identification data: (R & T and Grant numbers found on Page 1 of Grant)

   a. University of Maryland
   University Name

   b. N00014-94-1-0661
   Grant Number

   c. 333n013---01
   R & T Number

   d. Joel H. Saltz
   P.I. Name

   e. From: 01 Jul To: 30 Jun
   AASERT Reporting Period

NOTE: Grant to which AASERT award is attached is referred to hereafter as "Parent Agreement."

2. Total funding of the Parent Agreement and the number of full-time equivalent graduate students (FTEGS) supported by the Parent Agreement during the 12-month period prior to the AASERT award date.

   a. Funding: $ 66,854.40

   b. Number FTEGS: 1

3. Total funding of the Parent Agreement and the number of FTEGS supported by the Parent Agreement during the current 12-month reporting period.

   a. Funding: $ 0

   b. Number FTEGS: 0

4. Total AASERT funding and the number of FTEGS and undergraduate students (UGS) supported by AASERT funds during the current 12-month reporting period.

   a. Funding: $ 38,322

   b. Number FTEGS: 2

   c. Number UGS: 0

VERIFICATION STATEMENT: I hereby verify that all students supported by the AASERT award are U.S. citizens.

[illegible]
Principal Investigator

17 July 97
Date

19970804 067

REPORT DOCUMENTATION PAGE		Form Approved OMB No 0704 0188
[illegible]		
1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE 01 Jul 97	3. REPORT TYPE AND DATES COVERED Annual Technical, 01 Jul 96 - 30 Jun 97
4. TITLE AND SUBTITLE Scalable I/O for Irregular Loosely Synchronous Problems		5. FUNDING NUMBERS G N00014-94-1-0661
6. AUTHOR(S) Anurag Acharya, compiler		
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) University of Maryland Department of Computer Science College Park, Maryland 20742-3255		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Office of Naval Research, ONR 252 DG Ballston Tower One 800 North Quincy Street Arlington, VA 22217-5660		10. SPONSORING/MONITORING AGENCY REPORT NUMBER
11. SUPPLEMENTARY NOTES		
12a. DISTRIBUTION/AVAILABILITY STATEMENT		12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

To achieve good I/O performance on irregular, loosely synchronous problems it is necessary to work both at the application and the system support level. The first section describes our effor at developing an efficient out-of-core parallel sparse cholesky solver as an example of an irregular, loosely synchronous application and the second section describes the Jovian parallel I/O library which provides support for collective I/O operations.

14. SUBJECT TERMS I/O performance, parallel sparse cholesky solver, parallel library			15. NUMBER OF PAGES 2
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT

Report on *Scalable I/O for Irregular Loosely Synchronous Problems*

Joel Saltz, Anurag Acharya
Department of Computer Science
University of Maryland, College Park MD 20742

July 1997

In the first part of this project, we focused on performance of individual applications. We developed an out-of-core parallel sparse cholesky solver which achieved a maximum application-level I/O rate of 430 MB/s on our 16-processor IBM SP-2. This year, we have broadened our focus to cover a wider variety of applications. Our goal was to determine the requirements and techniques for a large class of applications.

To this end, we have analyzed a diverse suite of I/O-intensive parallel applications to determine their I/O requirements and the implications of these requirements for the design of I/O systems for parallel machines. We attempted to answer the following questions. First, what are the steady-state and peak I/O rates required by the application? These rates indicate how aggressive the I/O system needs to be. Second, what spatial patterns, if any, exist in the sequence of I/O requests? In particular, what are the common request sizes and whether I/O requests are sequential. Third, what is the degree of intra-processor and inter-processor locality in I/O accesses? These measures indicate whether caching previously accessed data is likely to improve performance and if so, where should the cache(s) be placed and what caching policies should be used. This information is also useful to understand the impact of alternative disk placements on application performance. For example, if all processors in an application access only their own partition of the data, the use of private disks instead of shared disks can reduce communication requirements and can increase aggregate I/O bandwidth. On the other hand, if the data in a partition is written by the owning processor but read by all processors, there may be no significant performance penalty to using shared disks. Fourth, does the application structure allow programmers to disclose future I/O requests to the I/O system? If this is indeed the case, it would provide an opportunity for the I/O system to improve the utilization of the storage devices as well as reduce the latency of I/O requests [?, ?]. Finally, what patterns, if any, exist in the sequence of inter-arrival times for I/O requests? This information would allow the I/O system to estimate when I/O requests that have been previously disclosed will actually be made. This can allow it to better schedule prefetches for the sequence of I/O requests disclosed by individual applications as well as those disclosed by a group of applications.

To address these questions, we have analyzed I/O request traces for a diverse set of I/O-intensive parallel applications. This set includes four non-scientific applications (IBM's DB2 Parallel Edition, datamining, a parallel web server and parallel textual search) and three parallel scientific applications. These applications have been tuned, to various degrees, to achieve good I/O performance. We believe that this is important as studying applications which have not been tuned for I/O performance can lead to misleading conclusions [?]. We ran these applications on an IBM SP-2 and captured the I/O requests using the `trace` facility provided by AIX 4.1. In addition, we have acquired the I/O request traces made available by the Pablo group at the University of Illinois, Urbana-Champaign [?]. These traces correspond to four parallel scientific applications from several domains.

Some of our conclusions were:

- Given the high steady-state and peak demands for read requests, I/O systems should be aggressive on optimizing data retrieval; we expect that simple write-behind policies would be effective given the low demand for writing data.
- For the current hardware, an I/O system that delivers a read bandwidth of about 19 MB/s per-processor should meet the requirements of even the most demanding applications and that a read bandwidth of 10 MB/s per-processor should be adequate for most applications.
- For the variety of applications examined in this study, request were usually large and the access patterns were both simpler and more complex than nested strides.
- There is little or no write-sharing between processors.
- Local disks are an important component of I/O systems for parallel machines.
- Different application require different caching policies – in particular, they need different cache replacement policies and different cache placement (server/client/both) decisions. Ideally, I/O systems for parallel machines should allow the application to control or specify the caching policy.
- For many applications, the sequence of inter-arrival times between read requests can be described by relatively simple patterns. We have seen three patterns: constant, piece-wise constant and piece-wise quadratic. The repetitve nature of the patterns suggests that it might be possible for the I/O system to determine the pattern during the first few repetitions and to use that information to estimate the inter-arrival times for the subsequent repetitions. We are planning to explore this further.